AFRL-RI-RS-TR-2010-123

HIGH-PERFORMANCE CLOCK SYNCHRONIZATION ALGORITHMS FOR DISTRIBUTED WIRELESS AIRBORNE COMPUTER NETWORKS WITH APPLICATIONS TO LOCALIZATION AND TRACKING OF TARGETS

Texas A&M University

June 2010

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■**UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2010-123 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
WALTER KOZIARZ	EDWARD J. JONES, Deputy Chief
Work Unit Manager	Advanced Computing Division
	Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
JUNE 2010

2. REPORT TYPE
Final

3. DATES COVERED *(From - To)*
May 2009 – January 2010

4. TITLE AND SUBTITLE
HIGH-PERFORMANCE CLOCK SYNCHRONIZATION ALGORITHMS FOR DISTRIBUTED WIRELESS AIRBORNE COMPUTER NETWORKS WITH APPLICATIONS TO LOCALIZATION AND TRACKING OF TARGETS

5a. CONTRACT NUMBER
N/A

5b. GRANT NUMBER
FA8750-09-1-0154

5c. PROGRAM ELEMENT NUMBER
63662D

6. AUTHOR(S)
Erchin Serpedin

5d. PROJECT NUMBER
WCNA

5e. TASK NUMBER
TA

5f. WORK UNIT NUMBER
MU

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Texas A&M University
Electrical & Computer Engineering Department
Tamus 3128
College Station, TX 77843-3128

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFRL/RITB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
N/A

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2010-123

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. PA# 88ABW-2010-3286
Date Cleared: 14-Jun-2010

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The goal of this project was to develop efficient synchronization schemes to ensure robust operation of wireless airborne networks in the absence of GPS (Global Positioning Systems), and in the presence of arbitrary network delay distributions. To cope with the Gaussian or non-Gaussian nature of the random network delays, a novel method, referred to as the Gaussian Mixture Kalman Particle Filter (GMKPF), is proposed to estimate the clock offset and shown to be robust to arbitrary network delays. GMKPF represents a better and more flexible alternative to the Gaussian Maximum Likelihood (GML), and Exponential Maximum Likelihood (EML) estimators for clock offset estimation in non-Gaussian or non-exponential random delay models. The computer simulations illustrate that GMKPF yields much more accurate results relative to GML and EML when the network delays are modeled in terms of a single non-Gaussian/non-exponential distribution or as a mixture of several distributions. As deliverables, the set of Matlab programs used to implement GMKPF and validate its performance are uploaded separately into Jiffy.

15. SUBJECT TERMS
Clock, estimation, phase, signal processing, skew, synchronization, wireless networks

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Walter Koziarz
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (*Include area code*)
U	U	U	UU	43	N/A

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

The PI would like to express his heartfelt thanks to Dr. Bruce Suter from Air Force Research Laboratory (AFRL), Rome, NY, for all his ideas, suggestions, support, and encouragement.

1. SUMMARY

The main goal of this project was to develop an efficient clock synchronization scheme to ensure robust operation of wireless airborne networks in conditions of arbitrary network delays and absence of GPS (Global Positioning Systems). To cope with the Gaussian or non-Gaussian nature of the random network delays, a novel method, referred to as the Gaussian Mixture Kalman Particle Filter (GMKPF), is proposed herein to estimate the clock offset in wireless sensor networks. GMKPF represents a better and more flexible alternative to the Gaussian Maximum Likelihood (GML), and Exponential Maximum Likelihood (EML) estimators for clock offset estimation in non-Gaussian or non-exponential random delay models. The computer simulations illustrate that GMKPF yields much more accurate results relative to GML and EML when the network delays are modeled in terms of a single non-Gaussian/non-exponential distribution or as a mixture of several distributions. As deliverables, the set of Matlab programs used to implement, simulate and validate the performance of GMKPF together with simulation results are provided.

2. INTRODUCTION

Future wireless airborne networks are envisioned to represent the next frontier of networking, to be pervasive and ubiquitous, and to provide a wide range of services and applications. This trend is underlined by a number of technological advances and demands. The rapidly growing demands for mobility and anywhere-anytime data access represent a major driving force behind the next generation of mobile wireless airborne networks. Recent technological developments mark also the departure of telecommunications systems from homogeneous networks to heterogeneous networks, from non-intelligent devices to smart devices, and from telephony-based services to multi-media services. In addition, recent advances in hardware and inexpensive wireless radio systems have made also possible the design of low-cost, low-power, and multi-functional sensor devices. When deployed in a large number across a geographical area, these sensor devices create a self-organized cooperative ad-hoc network that is perfectly fit for distributed sensing and automated information gathering, processing and communication. The upcoming years will very likely witness a growing demand for more intelligent sensor systems that will be networked with wireless local area networks (WLANs), Internet, satellite and Unmanned Aerial Vehicle (UAV) networks to create a global wireless airborne network with increased functionality and performance.

In general, for distributed computing and networking systems, maintaining the logical clocks of the computers in such a way that they are never too far apart is one of the most complex problems of computer engineering. Whether it is the disciplining of computer clocks with the devices synchronized to a Global Positioning System (GPS) satellite or a Network Time Protocol (NTP) time server over the Internet, it is possible to equip some primary time servers for the purpose of synchronizing a much larger number of secondary servers and clients connected through a common infrastructure. In order to do this, a distributed network clock synchronization protocol is required through which a server clock can be read, the readings to other clients can be transmitted and each client clock can be adjusted as required. In such a distributed synchronization approach, the participating devices exchange timing information with their chosen reference at regular intervals and adjust their logical clocks accordingly.

A computer clock in general has two components, namely a frequency source and a means of accumulating timing events (consisting of a clock interrupt mechanism and a counter implemented in software). The implementation of the computer clock in the operating system and the programming interface differ between operating systems and hardware platforms. However, the basic source of timing is an uncompensated quartz crystal oscillator and the clock interrupts it generates. Theoretically, two clocks would remain synchronized if their offsets are set equal and their frequency sources run at the same rate. However, practical clocks are set with limited precision and the frequency sources run at slightly different rates. In addition, the frequency of a crystal oscillator varies due to initial manufacturing tolerance, aging, temperature, pressure and other factors. Because of these inherent instabilities, distributed clocks must regularly be synchronized to keep them running close to each other.

The Network Time Protocol [22], [23] represents the most widely used clock synchronization protocol for large-scale networks with static topology such as the Internet. In NTP, the nodes are externally synchronized to a global reference time that is represented in the network by a set of master nodes or time servers that are referred to as layer-1 servers. The entire synchronization process assumes a hierarchical tree organization of the network nodes. Despite its wide-spread use in the synchronization of Internet, NTP is not appropriate for synchronization of wireless ad-hoc sensor networks that are subject to severe energy-constraints, dynamic topologies caused by mobility and node failures, and absence of GPS and global time references (due to either jamming, interferences, or absence of direct line of sight communication links). In addition, the service provided by NTP assumes continuous synchronization of all the network nodes with maximum accuracy and with no concern about energy consumption. However, NTP is not equipped with a mechanism to enable the local synchronization of a subset of nodes, and to keep the rest of the nodes switched to a power- saving (sleeping) state. Since listening continuously for the synchronization beacons is an energy-consuming operation, NTP cannot directly be applied to synchronization of energy-constrained wireless ad-hoc networks as is the case with wireless airborne networks and wireless sensor networks.

These considerations illustrate the need for novel distributed and scalable synchronization protocols for wireless ad-hoc networks that in general must satisfy a series of requirements: energy-efficiency, robustness with respect to node mobility and link/node failures, and ability to guarantee the long-term network synchronization at local and global scales.

Clock synchronization is important for many applications such as Internet delay measurements, cellular networks, data security algorithms, Media Access Control (MAC) protocols like Time Division Multiple Access (TDMA), Internet Protocol (IP) telephony, ordering of updates in database systems, etc. Recently, with the advent of Wireless Sensor Networks (WSNs) and Wireless Airborne Networks (WANs), developing clock synchronization protocols that suit their specific requirements is becoming an important research problem. A large number of their applications require the clocks of the nodes to run synchronously on a common timescale. This is the case with applications such as data fusion, efficient duty cycling operations, acoustic beamforming, localization, security and object tracking. Unlike conventional networks, energy efficiency must also be taken into account for addressing the clock synchronization problem in WSNs and WANs.

3. METHODS, ASSUMPTIONS AND PROCEDURES

During the last two decades, many clock synchronization protocols have been proposed such as [1], [2], [23], etc. NTP [23] is a protocol for synchronizing the clocks of computer systems over packet-switched, variable- latency data networks and it represents the Internet standard for time synchronization. It is a layered client- server architecture based on the User Data Protocol (UDP) message passing which synchronizes computer clocks in a hierarchical way using the offset delay estimation method. NTP's sender-receiver synchronization architecture is widely accepted in designing time synchronization algorithms and consists of the same two-way timing message exchange mechanism targeted in this project.

A protocol based on the remote clock reading method was put forward by [2], which handles unbounded message delays between processes. In [1], the time transmission protocol is used by a node to communicate the time on its clock to a target node, which subsequently estimates the time in the source node by using message timestamps and message delay statistics. For ad-hoc communication networks, the time synchronization protocol [8] represented one of the pioneering contributions in this area. The protocol is based on generating timestamps to record the time at which an event of interest occurred. The timestamps are updated by each node using its local clock and the time transformation method, where the final timestamp is expressed in terms of an interval with a lower bound and an upper bound. In the realm of wireless sensor networks, the clock synchronization protocols of particular note are Reference Broadcast Synchronization (RBS [5]), Timing Synch Protocol for Sensor Networks (TPSN [6]) and Time Diffusion Protocol (TDP [7]). RBS relies on simultaneous reception of broadcast pulses by several nodes transmitted by a common neighboring node after which the nodes exchange their timestamps and estimate the relative time offsets and skews. On the other hand, TPSN is based on the same sender-receiver paradigm as in NTP, like many other traditional clock synchronization protocols. The basic difference is that TPSN has been molded sufficiently to suit the requirements of wireless sensor networks. On the other side, TDP establishes a network-wide equilibrium time through an iterative, weighted averaging technique based on a diffusion of messages involving all the nodes in the synchronization process.

Clock synchronization between any two nodes is generally accomplished by message exchanges. Due to the presence of non-deterministic and possible unbounded message delays, messages can get delayed arbitrarily, which makes the clock synchronization very difficult [10]. The most commonly proposed non-deterministic network delay distributions are the Gaussian, exponential, Gamma, and Weibull probability density functions (pdfs) [9], [20], [25]. In general, it is difficult if not impossible to assess which distribution model may be fit to capture the network delay distributions in a given wireless sensor network (WSN). This is due to the fact that various factors might impact differently the distribution of network delays [17], [18]. The Gaussian pdf [12] and the exponential pdf [9] were also recently proposed to model the network delays in WSNs. Herein, the maximum likelihood (ML) estimators for clock offset estimation in the presence of Gaussian and exponential network delay distributions will be referred to as the Gaussian ML (GML) and exponential ML (EML), respectively. Reference [24] shows that GML and EML are quite sensitive to the network delay distributions. Therefore, one important problem that rises up is to design clock offset estimation schemes that are robust to the distribution of unknown network delays.

To overcome these challenges, in this project a novel clock offset estimation method, referred to as the Gaussian Mixture Kalman Particle Filter (GMKPF), is proposed and thoroughly tested. Extensive computer simulations illustrate GMKPF's merits of being robust and yielding very accurate clock offset estimates in the presence of arbitrary network delay distributions. The clock offset estimation framework adopted in this project is identical with the two-way message exchanges between two nodes, encountered in NTP [22] and TPSN protocol [24]. GMKPF combines the importance sampling (IS) based measurement update step with a KF (Kalman Filter) based Gaussian sum filter for the time-update and proposal density generation. Since GMKPF employs new observations and exploits the Expectation-Maximization (EM) algorithm to obtain the Gaussian Mixture Model (GMM), GMKPF is expected to exhibit better estimation performance when compared to GML and EML in general non-Gaussian/non-exponential delay models. Thus far, in the synchronization literature for WSNs, it appears that only very few preliminary and straightforward applications of standard Kalman filtering or general adaptive signal processing techniques were reported (see [14], [16] and [26]) to improve the mean square error (MSE) performance of protocols such as RBS [12] or TPSN [15].

In this project, upon designing the GMKPF, a thorough performance analysis of GMKPF, GML, and EML in the presence of the two-way message exchange mechanism between two nodes and symmetric/asymmetric Gaussian, exponential, Gamma, Weibull network delay distributions is first carried out. The performance of GMKPF, GML, and EML is also simulated under the mixing of two different distributions: Gaussian and exponential, Gaussian and Gamma, Gaussian and Weibull, exponential and Gamma, exponential and Weibull, and Gamma and Weibull delay distributions, respectively. The computer simulation results corroborate the superior performance of the proposed method relative to GML and EML, and its robustness to general network delay distributions. Therefore, the proposed GMKPF method represents a high-performance and very reliable clock offset estimation scheme fit to overcome the uncertainties caused by the network delay distributions.

4. RESULTS AND DISCUSSION

4.1. Problem Formulation and Objectives

The two-way timing message exchange mechanism is a recently proposed clock synchronization scheme for wireless sensor networks [15], [24]. Under this mechanism, the synchronization of two nodes A and B is achieved through a number of *N* cycles. Each cycle assumes two message transmissions: one from node A to node B, followed by a reverse transmission from node B to node A. At the beginning of the *k*th cycle, the node A sends its time reading $T_{1,k}$ to Node B, which records the arrival time of the message as $T_{2,k}$, according to its own time scale. Similarly a time message exchange is performed from Node B to Node A. At time $T_{3,k}$ node B transmits back to node A the time information $T_{2,k}$ and $T_{3,k}$. Denoting by $T_{4,k}$ the arrival time at node A of the message sent by node B, node A would then have access to the time information $T_{j,k}$, j = 1, . . . , 4 at the end of the *k*th cycle, which provide sufficient information for estimating the clock phase offset θ_A of node A relative to node B clock. The message exchanges that take place between two generic nodes A and B are depicted in Fig. 1.

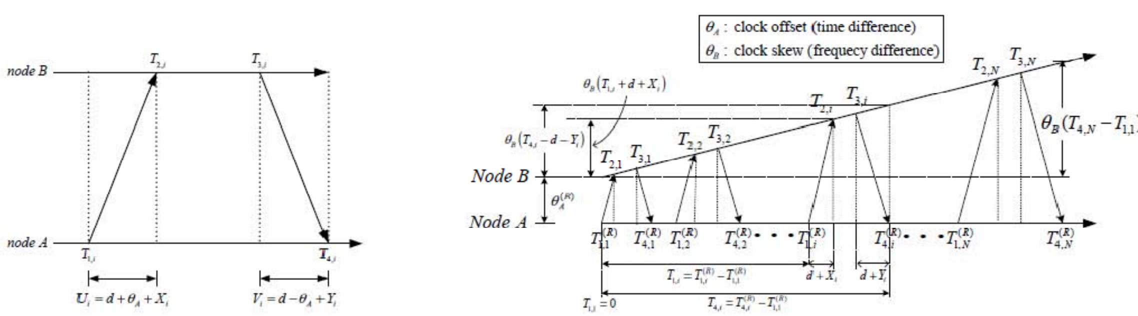

Figure 1. (a) A message exchange between two nodes A and B that present only clock phase offset. (b) Multiple message exchanges between nodes A and B that present clock phase offset and skew.

Utilizing the derivation presented in [24], the differences between the *k*th up and down-link delay observations corresponding to the *k*th timing message exchange are given by $U_k := T_{2,k} - T_{1,k} = d + \theta_A + L_k$ and $V_k = T_{4,k} - T_{3,k} = d - \theta_A + M_k$, respectively. The fixed value *d* denotes the fixed (deterministic) propagation delay component (which in general is neglected $d \approx 0$ in small range networks that assume RF transmissions). Parameters L_k and M_k stand for the variable portions of the network delays, and are assumed to be any distributions such as Gaussian, exponential, Gamma, Weibull or mixtures of two different distributions.

Given the observation samples $\mathbf{z}_k = [U_k, V_k]^T$, our goal is to find the minimum variance estimate of the unknown clock offset θ_A. For convenience, the notation $x_k := \theta_A$ will be used henceforth. Thus, it turns out that we are looking to determine the estimator:

$$\hat{x}_k = E\{x_k \mid \mathbf{Z}^l\}, \tag{1}$$

where $\mathbf{Z}^l$ denotes the set of observed samples up to time *l*, $\mathbf{Z}^l = \{\mathbf{z}_0, \mathbf{z}_1, \cdots, \mathbf{z}_l\}$. Since the clock offset value is assumed constant, the clock offset can be modeled as obeying a Gauss-Markov dynamic channel model of the form:

$$x_k = F x_{k-1} + v_{k-1}, \tag{2}$$

where F is the state transition matrix for clock offset. The additive noise component v_k can be modeled as Gaussian with zero mean and covariance $E\{v_k v_k^T\} = Q$. The vector observation model follows from the observed samples and it assumes the expression:

$$\mathbf{z}_k = \begin{bmatrix} d + x_k + L_k \\ d - x_k + M_k \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} d + \begin{bmatrix} 1 \\ -1 \end{bmatrix} x_k + \mathbf{n}_k, \tag{3}$$

where the observation noise vector $\mathbf{n}_k = [L_k, M_k]^T$ accounts for the random delays. One can now observe that eqs. (2) and (3) recast our initial clock offset estimation problem into a Gauss-Markov estimation problem with unknown states.

4.2. A Composite Particle Filtering Approach

Particle filtering is a sequential Monte Carlo sampling method built within the Bayesian paradigm. From a Bayesian perspective, at time *k*, the posterior distribution $p(x_k \mid \mathbf{z}_{0:k})$ is the main entity of interest. However, due to the non-Gaussianity of the model (3), the analytical expression of $p(x_k \mid \mathbf{z}_{0:k})$ cannot be obtained in closed-form expression, excepting for some special cases like Gaussian or exponential pdfs. Alternatively, particle filtering can be applied to approximate $p(x_k \mid \mathbf{z}_{0:k})$ by stochastic samples generated using a sequential importance sampling strategy.

Since the particle filtering with the prior importance function employs no information from observations in proposing new samples, its use is often ineffective and leads to poor filtering performance. Herein, we implement a slightly changed version of the Gaussian Mixture Sigma Point Particle Filter (GMSPPF) proposed in [21], and which will be referred to as a composite approach. This composite approach comes out from the utilization of another filtering technique producing a filtering probability density function used as importance function (IF) for the particle filtering.

The GMSPPF is a family of methodologies that use hybrid sequential Monte Carlo simulation and a Gaussian sum filter to efficiently estimate posterior distributions of unknown states in a non-linear dynamic system. However, in our state space modeling, because of the linear model, we do modify this method further. Following [21], we will next describe briefly the general framework assumed by the GMKPF method, obtained by replacing the SPKF with a KF. We next outline the main features of the proposed approach. First, we remark that any probability density $p(x)$ can be approximated as closely as desired by a Gaussian mixture model (GMM) of the following form [11],

$$p(x) \approx p_g(x) = \sum_{g=1}^{G} \alpha^{(g)} \mathbf{N}(x; \mu^{(g)}, P^{(g)}), \qquad (4)$$

where G stands for the number of mixing components, $\alpha^{(g)}$ denote the mixing weights and $\mathbf{N}(x; \mu, P)$ is a normal distribution with mean μ and covariance P. Thus, the predicted and updated Gaussian components, i.e., the means and covariances of the involved probability densities (posterior, importance, and so on) are calculated using the Kalman filter (KF) instead of the Sigma Point Kalman Filter (SPKF) [19], [21]. Since the state and observation equations are

linear, the KF was employed instead of the SPKF. Therefore, the resulting approach is called the Gaussian mixture Kalman particle filter (GMKPF). In order to avoid the particle depletion problem in cases where the observation (measurement) likelihood is very peaked, the GMKPF represents the posterior density by a GMM which is recovered from the re-sampled equally weighted particle set using the Expectation-Maximization (EM) algorithm.

In general for the particle filtering approach, the posterior density $p(x_{0:k} \mid \mathbf{z}_{1:k})$, where $x_{0:k} = \{x_0, \cdots, x_k\}$ and $\mathbf{z}_{1:k} = \{\mathbf{z}_1, \cdots, \mathbf{z}_k\}$, constitutes the complete solution to the sequential estimation problem. Our objective is to generate samples from the distribution $p(x_{0:k} \mid \mathbf{z}_{1:k})$. For this purpose, we have collected N sets of samples $x_{0:k}^{(i)} = \{x_0^{(i)}, \cdots, x_k^{(i)}\}$ with weights $w_k^{(i)}$, $i = 1, \ldots, N$. The particles $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{N}$ approximate $p(x_{0:k} \mid \mathbf{z}_{1:k})$. Finally, the conditional mean state and the corresponding error covariance can be calculated:

$$\overline{x}_k = \sum_{i=1}^{N} w_k^{(i)} x_k^{(i)}, \Phi = \sum_{i=1}^{N} w_k^{(i)} [\overline{x}_k - x_k^{(i)}][\overline{x}_k - x_k^{(i)}]^T . \tag{5}$$

At the end of each recursion, the particles are resampled to ensure they occur with the same probability as the weights.

The GMKPF combines the importance sampling (IS) based measurement update step with a KF based Gaussian sum filter for the time-update and proposal density generation. In the time update stage, GMKPF approximates the prior, proposal and posterior density function as GMMs using banks of parallel KFs. The updated mean and covariance of each mixand follow from the KF updates. In the measurement update stage, the GMKPF uses a finite GMM representation of the posterior filtering density

$$p_g(x_k \mid z_k) = \sum_{g=1}^{G} \alpha_k^{(g)} N(x_k; \mu_k^{(g)}, P_k^{(g)}) , \tag{6}$$

where G is the number of GMMs, $\alpha_l^{(g)}$ are the mixing weights and $N(x_k; \mu_k^{(g)}, P_k^{(g)})$ is a normal distribution determined from the gth KF with predicted mean $\mu_k = \overline{x}_k$ and positive definite covariance P_k. This is recovered from the weighted posterior particle set of the IS based measurement update stage, by means of an Expectation-Maximization (EM) [13] step. The EM algorithm can be used to obtain Gaussian Mixture approximations from these particles and

weights. Through this mechanism, the EM-based posterior GMM further mitigates the "sample depletion" problem through its inherent "kernel smoothing" nature. The EM algorithm provides an iterative method to estimate $\bar{\theta}$ via

$$\bar{\theta} = \arg\max_{\theta} p(\mu \mid \theta) , \tag{7}$$

with the Gaussian mixture specified by the parameter set $\theta = \{\alpha_l^{(1)}, \cdots, \alpha_l^{(G)}, \mu_l^{(1)}, \cdots, \mu_l^{(G)}, P_l^{(1)}, \cdots, P_l^{(G)}\}$. Specifically, the EM algorithm is a two-step iterative algorithm which works as follows: given a $\theta^{(j)}$, it finds the next value $\theta^{(j+1)}$ via

- E-step: $Q(\theta \mid \theta^{(\mathbf{j})}) = E\{\log p(\mu \mid \theta) \mid \theta^{(\mathbf{j})}\}$
- M-step: $\theta^{(j+1)} = \arg\max_{\theta} Q(\theta \mid \theta^{(j)})$

The reader is directed to reference [13] for more detailed explanations of the EM algorithm for GMM. Finally, the conditional mean state estimate and the corresponding error covariance can be calculated as follows:

$$\bar{x}_k = \sum_{g=1}^{G} \alpha_k^{(g)} \mu_l^{(g)},\ \bar{P}_k = \sum_{g=1}^{G} \alpha_k^{(g)} [P_k^{(g)} + (\mu_k^{(g)} - \bar{x}_k)(\mu_k^{(g)} - \bar{x}_k)^T] \tag{8}$$

Below we provide a fairly pseudo-code for a GMKPF algorithm that is fit for estimating clock offsets in non-Gaussian delay models.

Algorithm

(1) At time k-1, initialize the densities

- The posterior density is approximated by

$$p_g(x_{k-1} \mid \mathbf{z}_{k-1}) = \sum_{g=1}^{G} \alpha_{k-1}^{(g)} \mathbf{N}(x_{k-1}; \mu_{k-1}^{(g)}, P_{k-1}^{(g)})$$

- The process noise density is approximated by

$$p_g(v_{k-1}) = \sum_{i=1}^{I} \beta_{k-1}^{(i)} \mathbf{N}(v_{k-1}; \mu_{v_{k-1}}^{(i)}, Q_{k-1}^{(i)})$$

- The observation noise density is approximated by

$$p_g(\mathbf{n}_k) = \sum_{j=1}^{J} \gamma_k^{(J)} \mathbf{N}(\mathbf{n}_k; \mu_{\mathbf{n}_k}^{(j)}, R_k^{(j)})$$

(2) Pre-prediction step

- Calculate the pre-predictive state density using KF, $\tilde{p}_g(x_k \mid \mathbf{z}_{k-1})$
- Calculate the pre-posterior state density using KF, $\tilde{p}_g(x_k \mid \mathbf{z}_k)$

(3) Prediction step

- the predictive state density using GMM, $\hat{p}_g(x_k \mid \mathbf{z}_{k-1})$
- Calculate the posterior state density using GMM, $\hat{p}_g(x_k \mid \mathbf{z}_k)$

(4) Observation Update step

- Draw N samples $\{\chi_k^{(l)}; l = 1, \cdots, N\}$ from the importance density function, $q(x_k \mid \mathbf{z}_k) = \hat{p}_g(x_k \mid \mathbf{z}_k)$
- Calculate their corresponding importance weights:

$$\tilde{w}_k^{(l)} = \frac{p(\mathbf{z}_k \mid \chi_k^{(l)})\,\hat{p}_g(\chi_k^{(l)} \mid \mathbf{z}_{k-1})}{\hat{p}_g(\chi_k^{(l)} \mid \mathbf{z}_k)}$$

- Normalize the weights: $w_k^{(l)} = \tilde{w}_k^{(l)} / \sum_{l=1}^{N} \tilde{w}_k^{(l)}$
- Approximate the state posterior distribution using the EM-algorithm, $p_g(x_k \mid \mathbf{z}_k)$

(5) Infer the conditional mean and covariance:

- $\overline{x}_k = \sum_{l=1}^{N} w_k^{(l)} \chi_k^{(l)}$ and $\overline{P}_k = \sum_{l=1}^{N} w_k^{(l)} (\chi_k^{(l)} - \overline{x}_k)(\chi_k^{(l)} - \overline{x}_k)^T$
- Or equivalently, upon fitting the posterior GMM, calculate the variables in eq. (8).

4.3. Implementation Aspects of GMKPF Algorithm

GMKPF can be viewed as an efficient tool to perform probabilistic inference, i.e., to estimate the hidden variables (states or parameters) of a system in an optimal and consistent fashion given noisy or incomplete observations. Fig. 2 depicts a general framework for probabilistic inference.

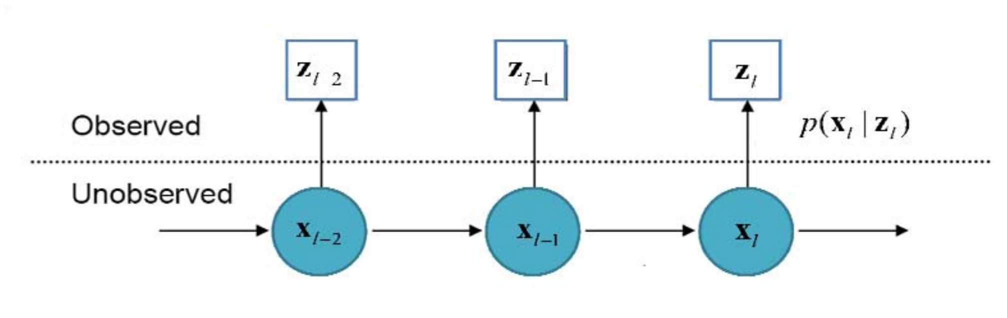

Figure 2. Probabilistic Inference

A block diagram of GMKPF is represented in Fig. 3.

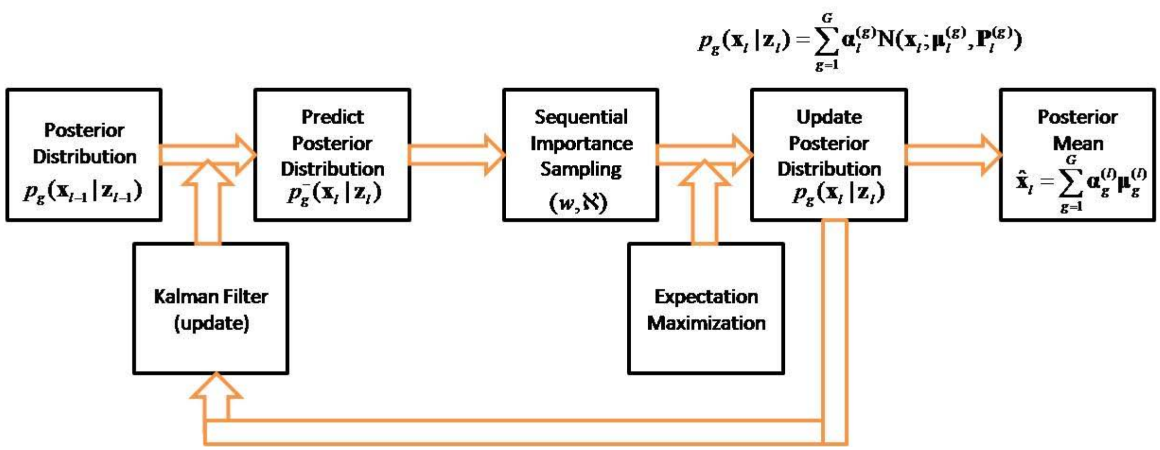

Figure 3. Constitutive Blocks of Gaussian Kalman Particle Filter

The Matlab functions and routines that have been used for implementing the GMKPF algorithm are depicted in Figs. 4 and 5.

Main function

Asymmetric Exponential : main_gmm_asym_expo.m and gssm_asym_expo.m
Asymmetric Gaussian : main_gmm_asym_gauss.m and gssm_asym_gauss.m
Symmetric Exponential : main_gmm_expo.m and gssm_expo.m
Symmetric Gaussian : main_gmm_gauss.m and gssm_gauaa.m
Gamma : main_gmm_gamma.m and gssm_gamma.m
Weibull : main_gmm_wei.m and gssm_wei.m

Common function

GMMFIT : Fit a Gaussian mixture model (GMM) with M components to dataset X using an expectation maximization algorithm (EM)
GMMPROBABILITY : Calculates any of the related (through Bayes rule) probabilities of a Gaussian Mixture Model (gmmDS) and a given dataset X. 'prob_type' is a string indicating which of the four probability

```
%
%                  P(X|C) . P(C)                    likelihood . prior
%     P(C|X) = -----------------     posterior = --------------------
%                      P(X)                              evidence
%
```

GMMINITIALIZE : Initialises Gaussian mixture model (GMM) from data
KMEANS : Trains a k means cluster model. Use the EM algorithms.
GMMSAMPLE : Draw N samples from the Gaussian mixture model (GMM) described by the GMM data structure .
SRCDKF : Square Root Central Difference Kalman Filter (Sigma-Point Kalman Filter variant) (In linear model, almost same as Kalman Filter)

Figure 4. List of MATLAB functions used to implement the Gaussian Mixture Kalman Particle Filter

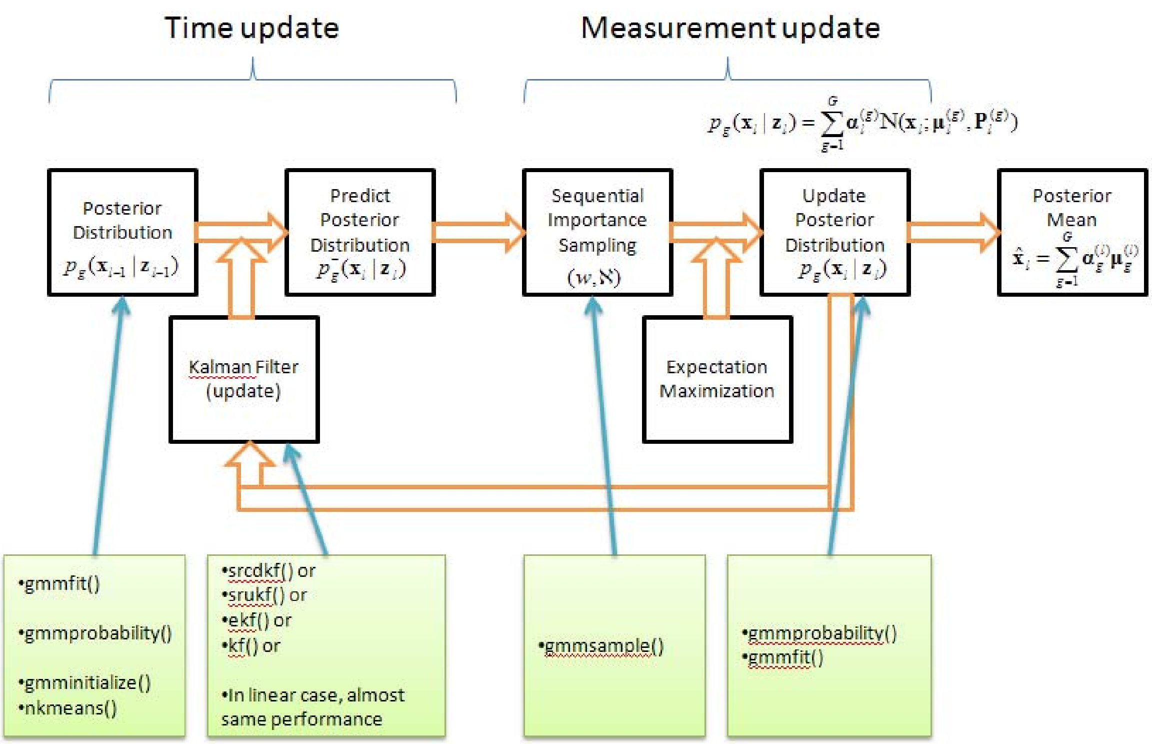

Figure 5. Main MATLAB Functions used to implement the Gaussian Mixture Kalman Particle Filter

4.4. General Simulation Results

In this section, computer simulation results will be offered to assess the performance of GMKPF, GML [24], and EML [24]-approaches for estimating the clock offset in wireless sensor networks. We consider a total of 10 delay models: asymmetric Gaussian, exponential, Gamma, Weibull, and mixtures of Gaussian and exponential, Gaussian and Gamma, Gaussian and Weibull, exponential and Gamma, exponential and Weibull, and Gamma and Weibull distributions. The reason for this study is to illustrate that the proposed method is robust, exhibits superior performance and can be applied to deal with any delay distribution. The stationary process v_k is assumed to achieve a given constant variance $Q = 1e - 4$. The number of particles and GMM are 100 and 3, respectively.

Figs. 6-9 show the MSE (Mean Square Error) of the estimators assuming that the random delay models are asymmetric Gaussian, exponential, Gamma, Weibull pdfs, respectively. The subscripts 1 and 2 are used to differentiate the parameters of delay distributions corresponding to uplink and downlink, respectively. As an example, the parameters σ_1 and σ_2 in Fig. 6 denote the standard deviations of uplink and downlink asymmetric Gaussian network delay densities, respectively. The MSEs are plotted against the number of observations, ranging from 5 to 25. Note that the GMKPF performs much better (a reduction of MSE with over 100%) when compared to GML or EML. It is interesting to note that the MSE of GML exhibits better performance than EML in the asymmetric Gaussian delay model case and poorer performance in the presence of asymmetric exponential, Gamma, and Weibull delay models. The reason for this is that Gamma and Weibull delay models are closer to the exponential distribution than the Gaussian distribution.

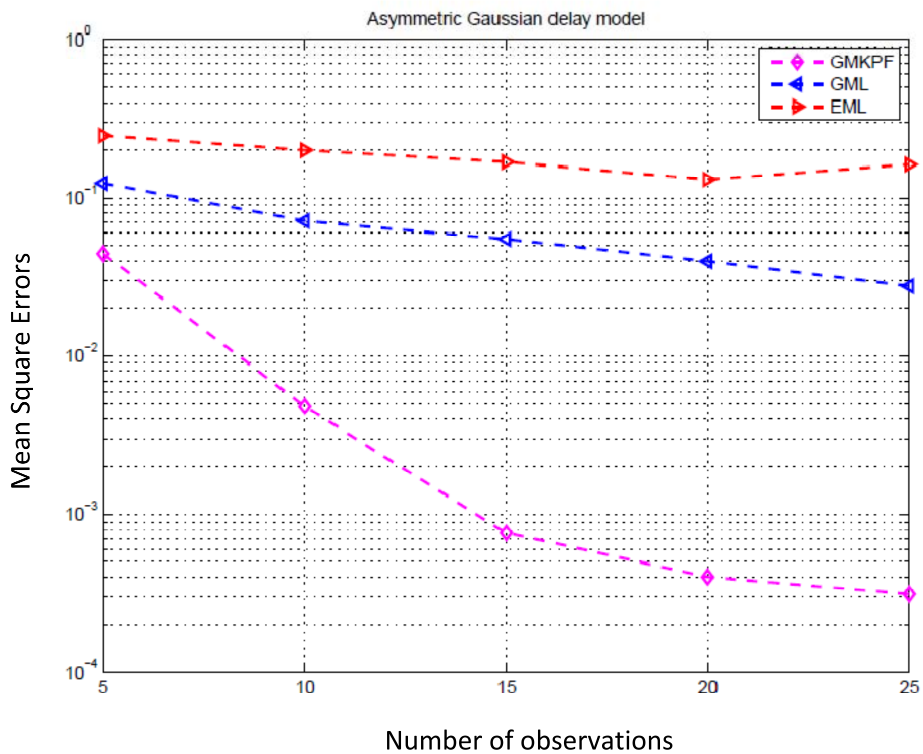

Figure 6. MSEs of clock offset estimators for asymmetric Gaussian random delays (σ_1=1, σ_2=4)

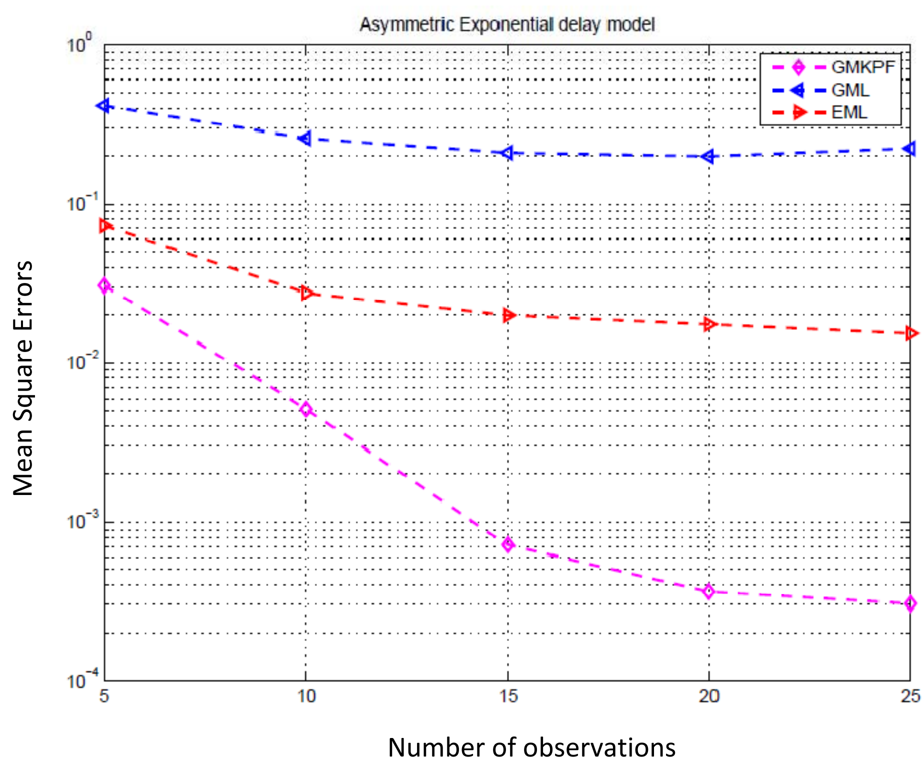

Figure 7. MSEs of clock offset estimators for asymmetric Exponential random delays (λ_1=1, λ_2=5)

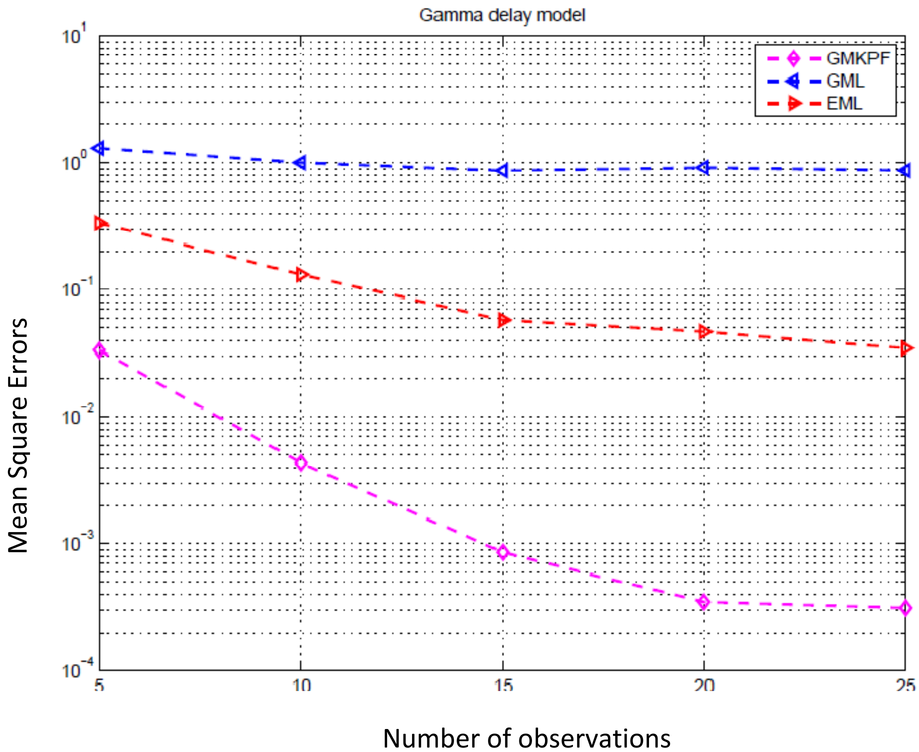

Figure 8. MSEs of clock offset estimators for Gamma random delays (α_1=2, β_1=1)

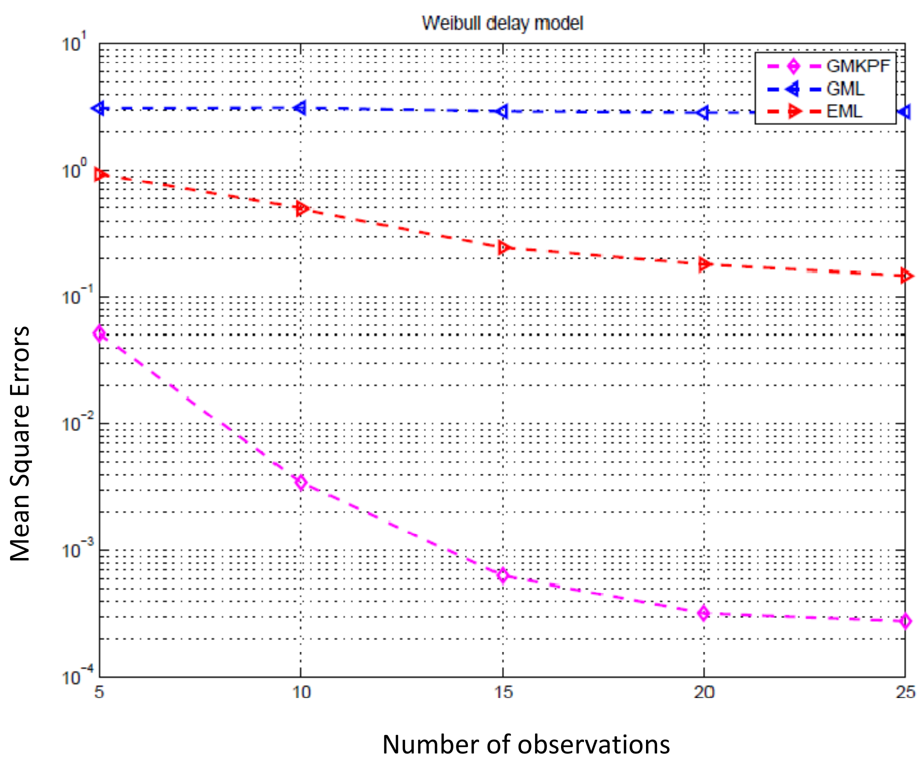

Figure 9. MSEs of clock offset estimators for Weibull random delays (α_1=2, β_1=2) and (α_2=6, β_2=2)

To further quantify the robustness of the estimators, we studied the performance of the GMKPF, GML, and EML under various network delay conditions, where the random delay models are mixtures of two distributions. For examples, in Fig. 10, we mix equally a Gaussian with an exponential delay model, each having a weight of 50%. This means that if 10 observations are received, 5 observations are Gaussian and the remaining 5 samples assume an exponential distribution. From Figs. 10-15, we observe that GMKPF clearly outperforms the GML and EML. In these cases, the GML presents better performance than EML if the network delay process is closer to a Gaussian. Otherwise, the EML exhibits better performance than the GML, while GMKPF outperforms both the GML and EML.

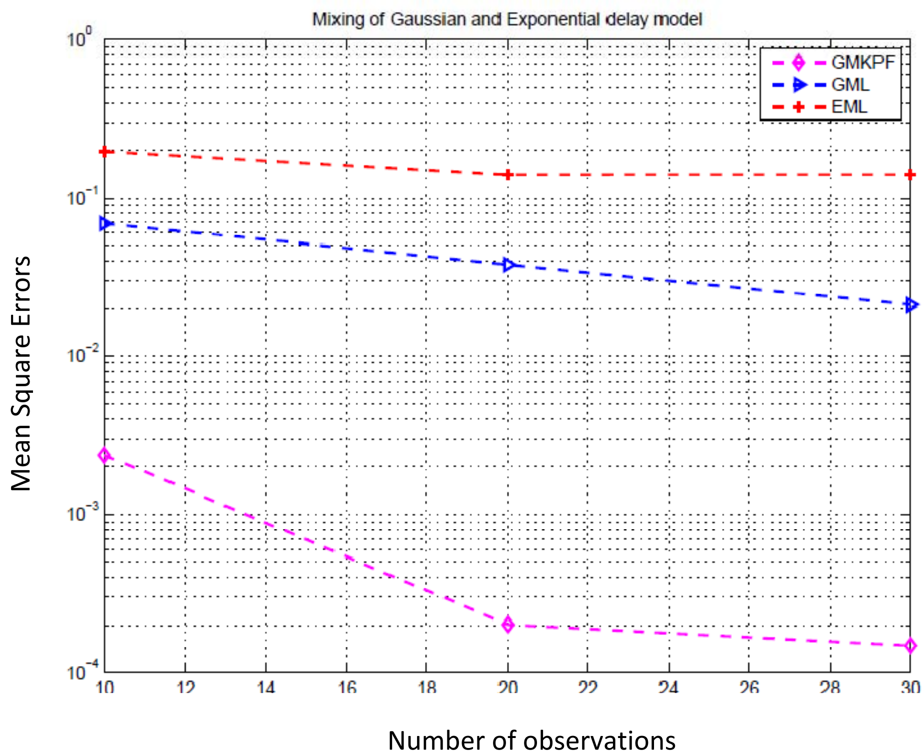

Figure 10. MSEs of clock offset estimators for mixing of a Gaussian (σ_1=1, σ_2=1) and an Exponential (λ_1=1, λ_2=5)

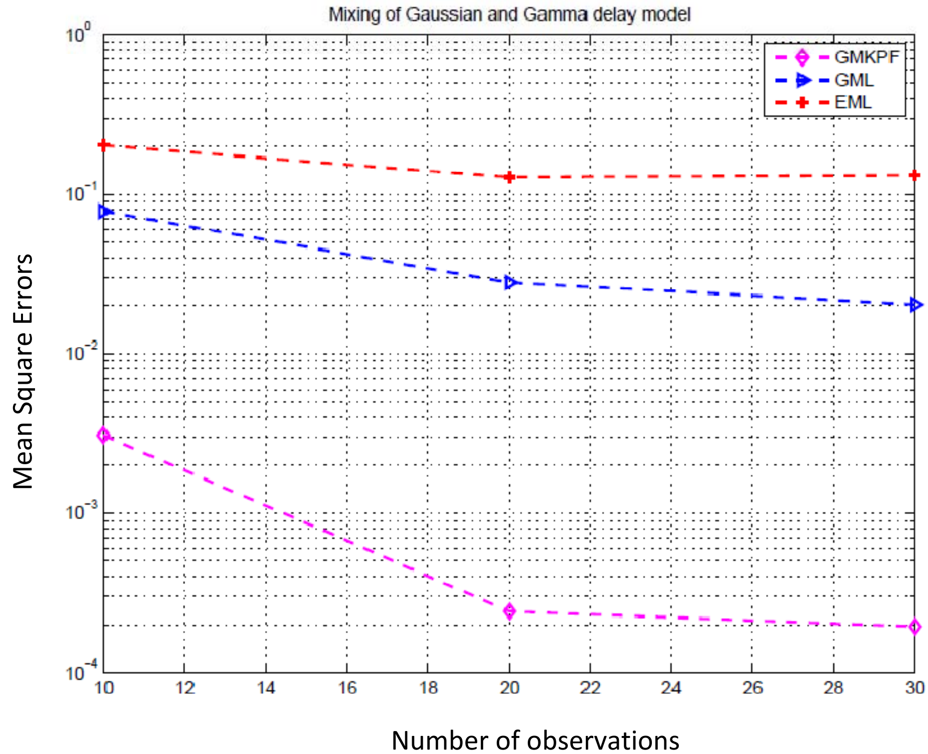

Figure 11. MSEs of clock offset estimators for mixing a Gaussian (σ_1=1, σ_2=1) and a Gamma (α_1=2, β_1=2)

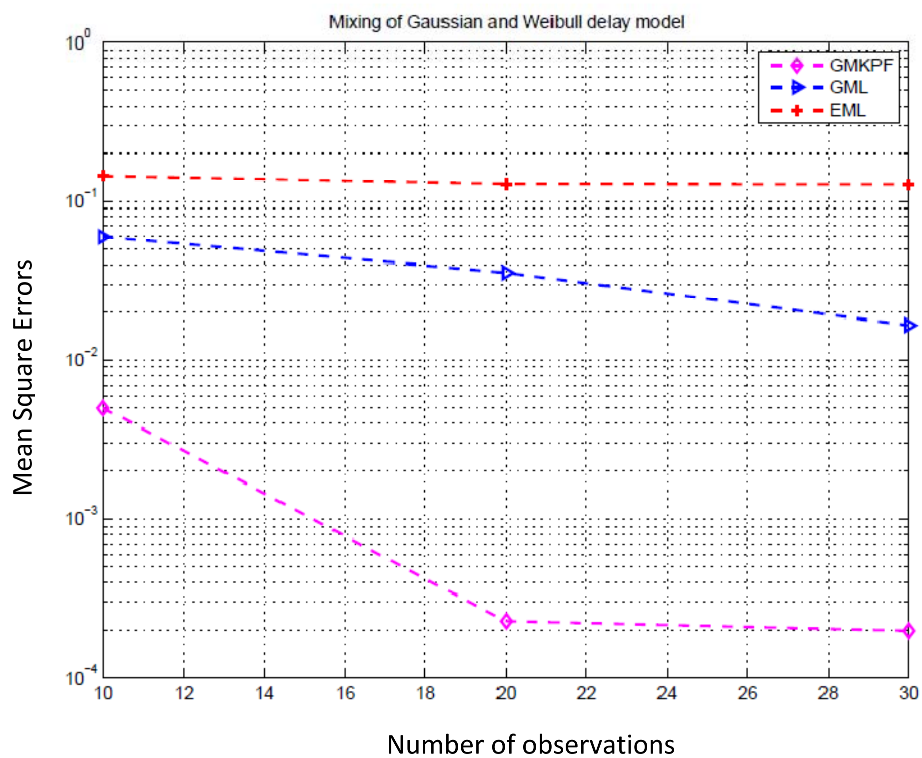

Figure 12. MSEs of clock offset estimators for mixing a Gaussian (σ_1=1, σ_2=1) and a Weibull random delay (α_1=2, β_1=2 and α_2=6, β_2=2)

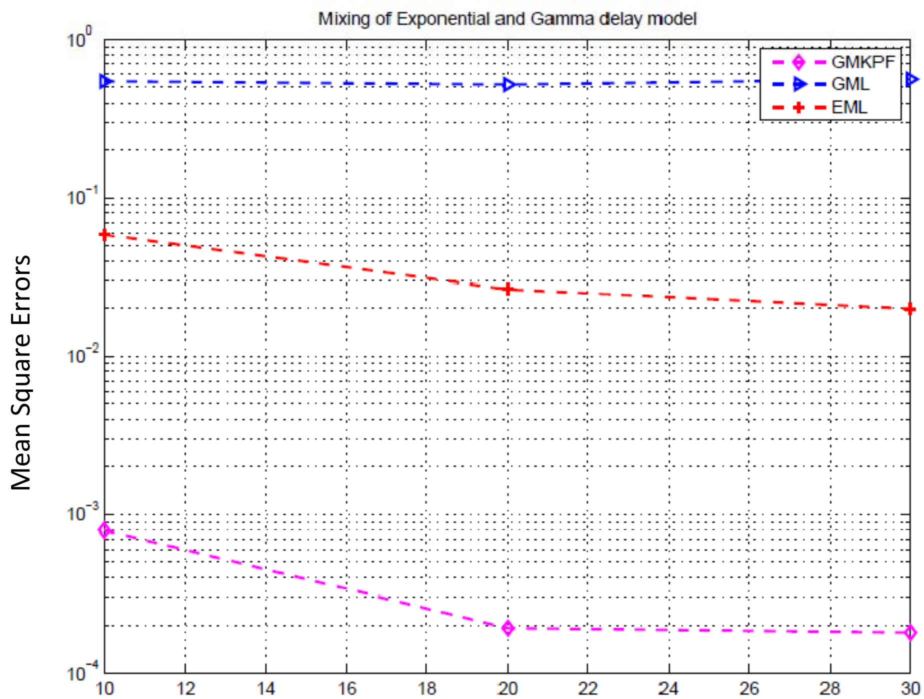

Figure 13. MSEs of clock offset estimators for mixing an Exponential (λ_1=1, λ_2=5) and a Gamma (α_1=2, β_1=5 and α_2=2, β_2=2)

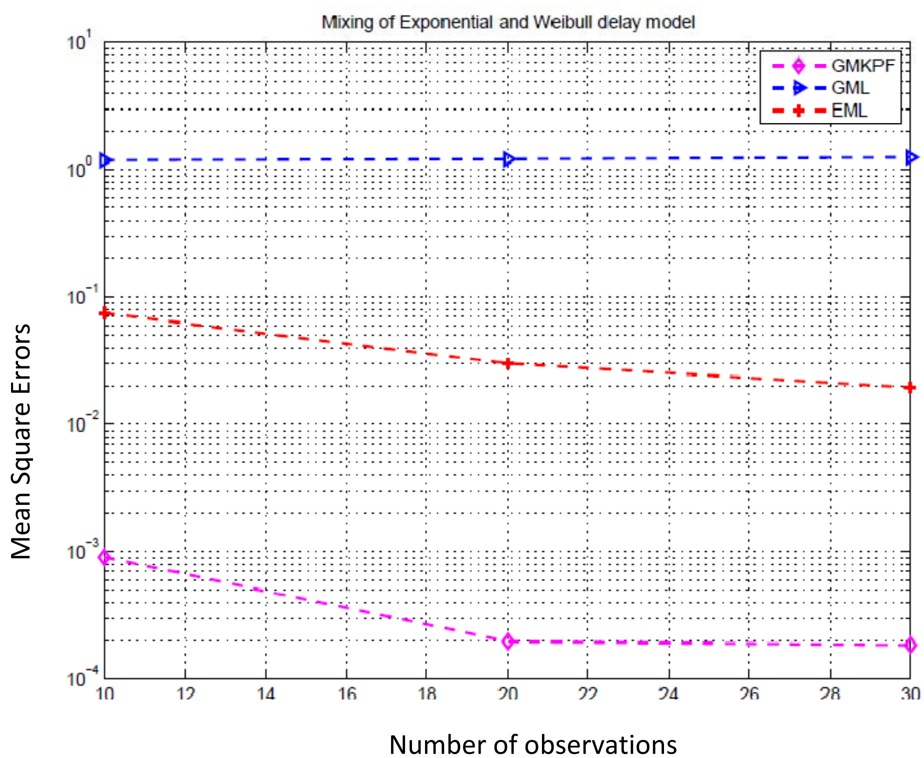

Figure 14. MSEs of clock offset estimators for mixing an Exponential (λ_1=1, λ_2=5) and a Weibull (α_1=2, β_1=2 and α_2=6, β_2=2)

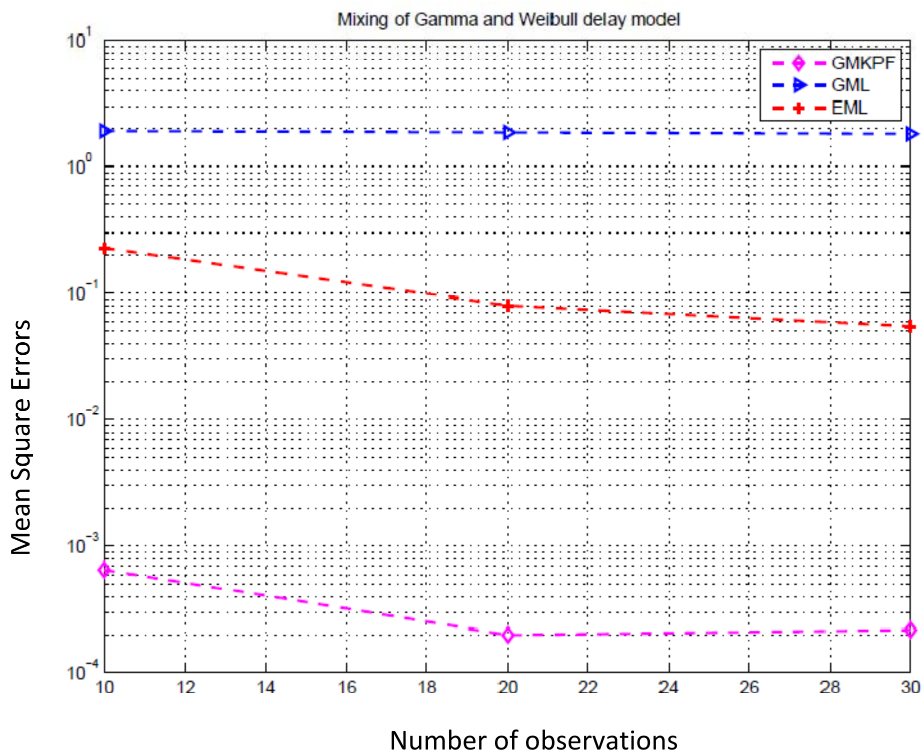

Figure 15. MSEs of clock offset estimators for mixing a Gamma (α_1=2, β_1=5 and α_2=2, β_2=2) and a Weibull (α_1=2, β_1=2 and α_2=6, β_2=2)

4.5. In-Depth Assessment of GMKPF Performance

- TPSN - clock offset model (1): (Asymmetric Gaussian)

  GMKPF conditions: Initial value: exponential ML value, 3-component GMM, 500 particles, process noise variance: 1e-6, Monte Carlo Simulations: 200.

  The MSEs of clock offset estimators in the presence of asymmetric Gaussian delays are presented in Fig. 16.

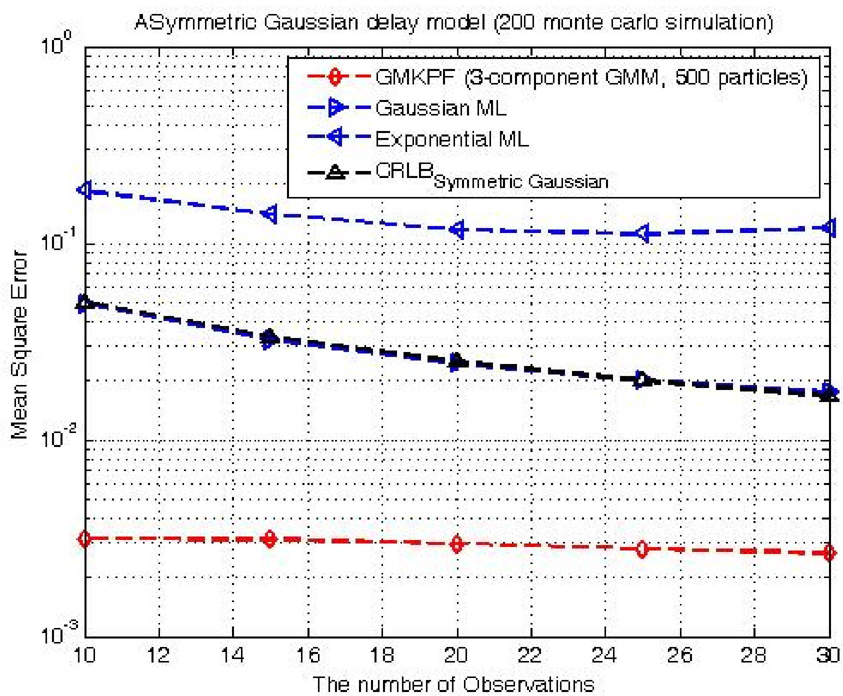

Figure 16. MSEs of clock offset estimators for asymmetric Gaussian delays

- TPSN - clock offset model (2): (Asymmetric Exponential)

  GMKPF conditions: Initial value: exponential ML value, 3-component GMM, 500 particles, process noise variance: 1e-6. Monte Carlo Simulations: 200.

  The MSEs of clock offset estimators in the presence of asymmetric Exponential delays are presented in Fig. 17.

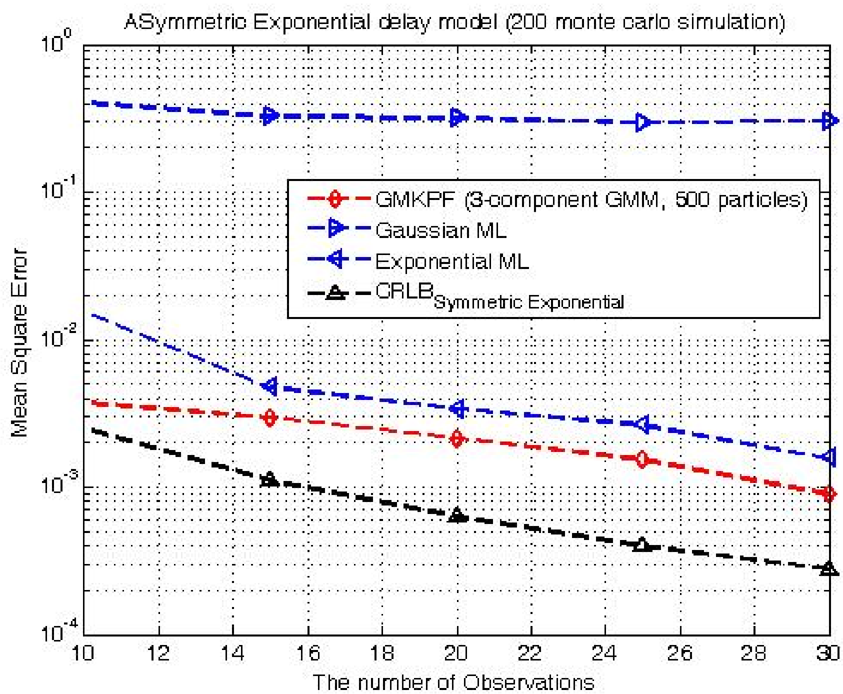

Figure 17. MSEs of clock offset estimators for asymmetric Exponential delays

- TPSN - clock offset model (3): (Gamma)

  GMKPF conditions: Initial value: exponential ML value, 3-component GMM, 500 particles, process noise variance: 1e-6. Monte Carlo Simulations: 200.

  The MSEs of clock offset estimators in the presence of asymmetric Gamma delays are presented in Fig. 18.

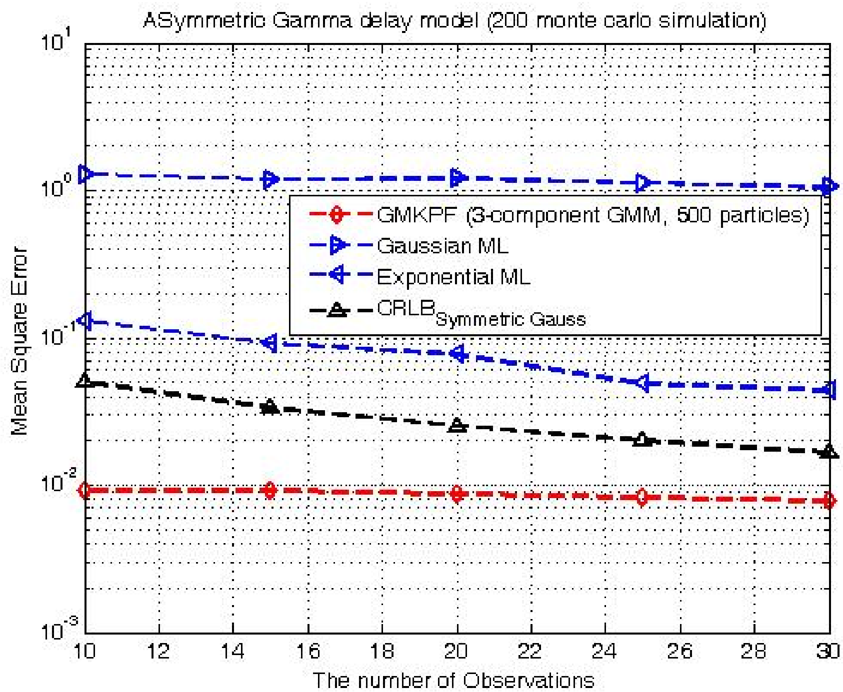

Figure 18. MSEs of clock offset estimators for asymmetric Gamma delays

- TPSN - clock offset model (4): (Weibull)

  GMKPF conditions: Initial value: exponential ML value, 3-component GMM, 500 particles, process noise variance: 1e-6. Monte Carlo Simulations: 200.

  The MSEs of clock offset estimators in the presence of asymmetric Weibull delays are presented in Fig. 19.

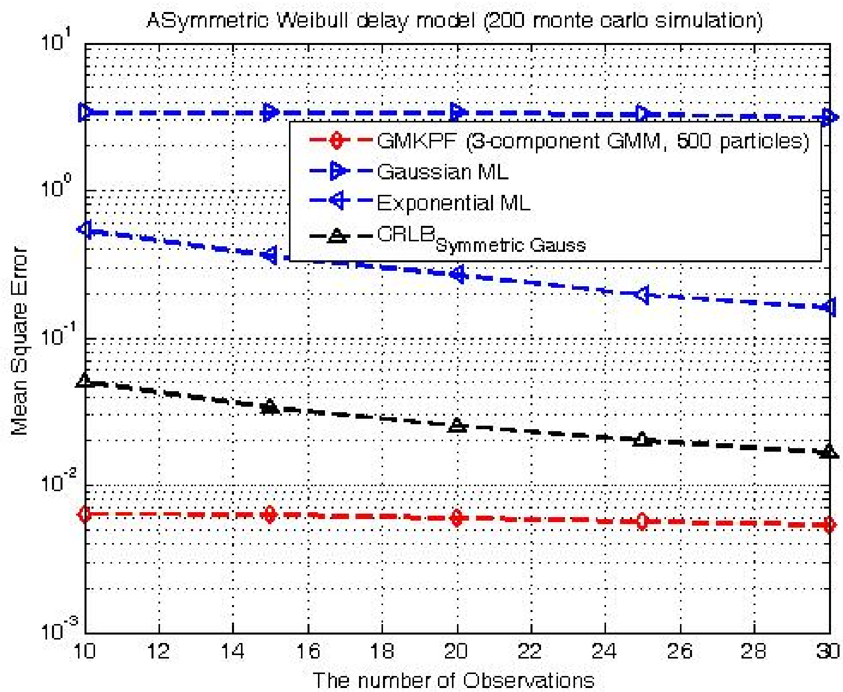

Figure 19. MSEs of clock offset estimators for asymmetric Weibull delays.

- TPSN - clock offset model (1): (Gamma)

  GMKPF conditions: Initial value: exponential ML value, 3, 5, 7-component GMM, 500 particles, process noise variance: 1e-6. Monte Carlo Simulations: 300.

  The MSEs of clock offset estimators in the presence of asymmetric Gamma delays are presented in Fig. 20.

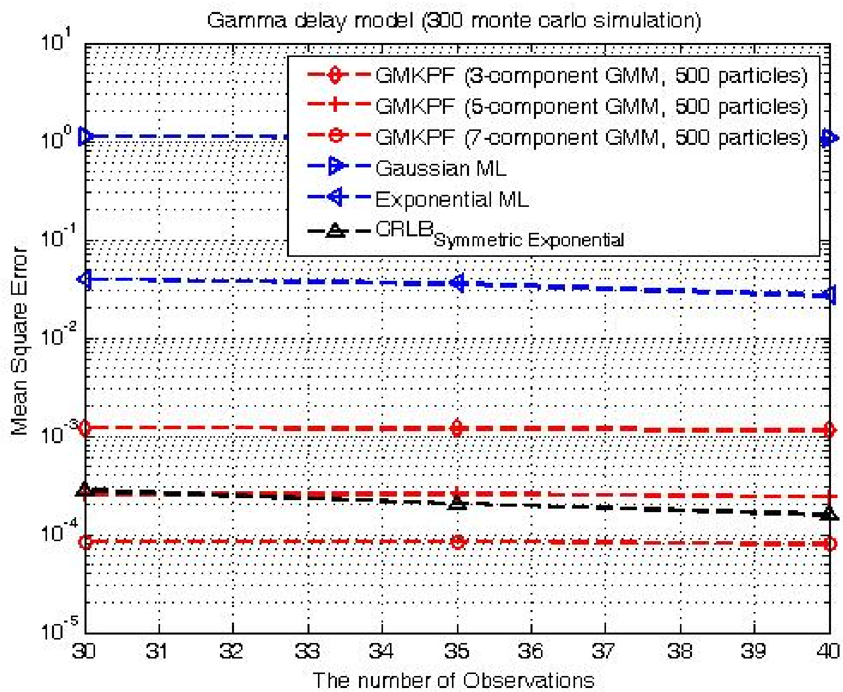

Figure 20. MSEs of clock offset estimators for asymmetric Gamma delays.

- TPSN - clock offset model (2): (Gamma)

  The total number of message exchanges in a network with 100 nodes and MSE=0.001 are as follows:

  $$N_{TPSN-EML} = 2N(L-1) = 2\times 920\times(100-1) = 182160$$

  $$N_{TPSN-GMKPF} = 2N(L-1) = 2\times 30\times(100-1) = 5940$$

  The number of timing messages vs. MSEs of clock offset estimators in the presence of asymmetric Gamma delays are presented in Fig. 21. Monte Carlo Simulations: 400.

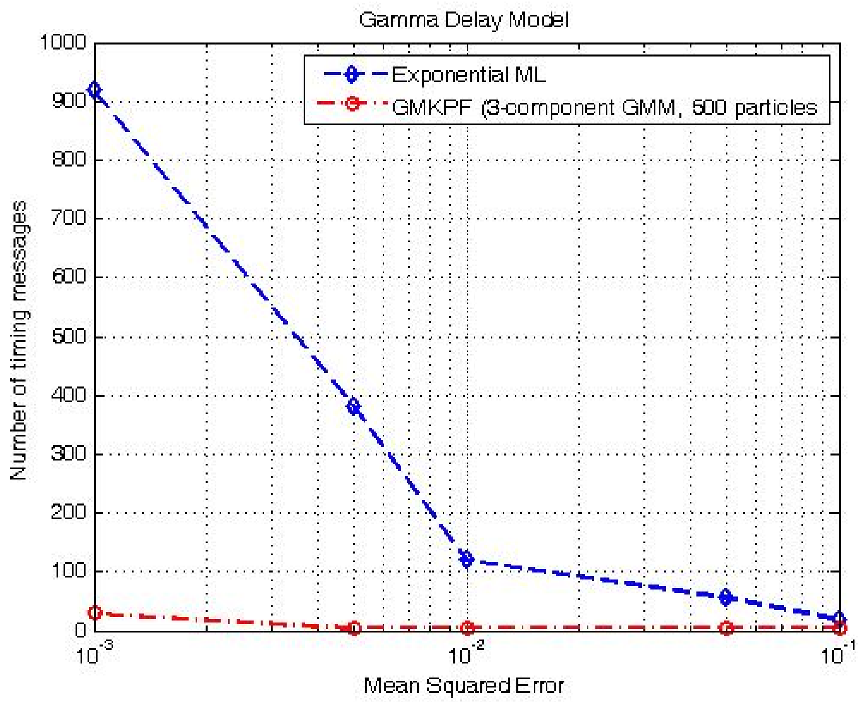

Figure 21. Number of timing messages vs. MSEs of clock offset estimators for asymmetric Gamma delays

- TPSN - clock offset model (1): (Weibull)

  GMKPF conditions: Initial value: exponential ML value, 3, 6, 10-component GMM, 500 particles, process noise variance: 1e-6. Monte Carlo Simulations: 300.

  The MSEs of clock offset estimators in the presence of asymmetric Weibull delays are presented in Fig. 22.

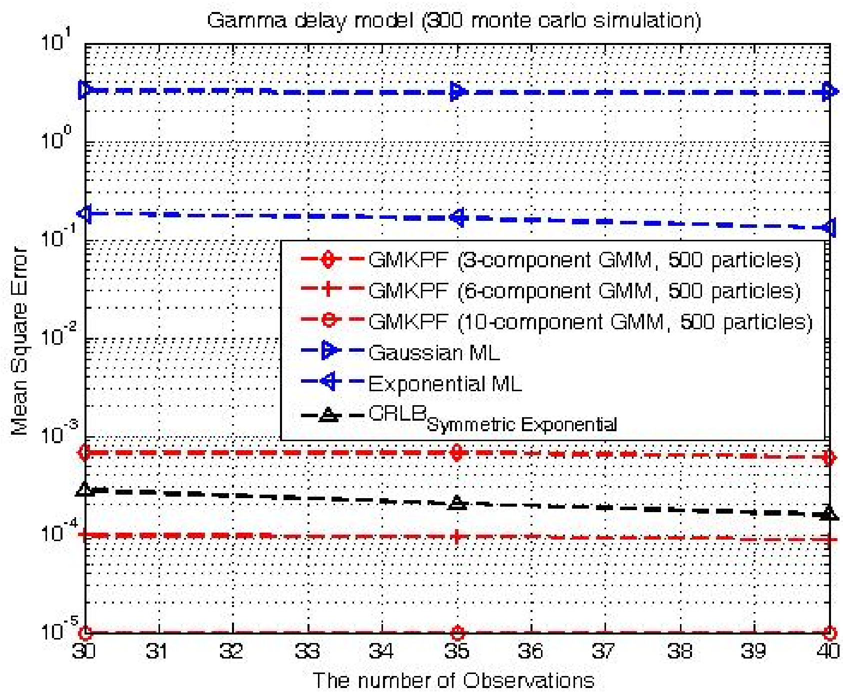

Figure 22. MSEs of clock offset estimators for asymmetric Weibull delays

- TPSN - clock offset model (2): (Weibull)

  The total number of message exchanges in a network with 100 nodes and MSE=0.001 are as follows:

  $N_{TPSN-EML} = 2N(L-1) = 2 \times 220 \times (100-1) = 399960$

  $N_{TPSN-GMKPF} = 2N(L-1) = 2 \times 30 \times (100-1) = 5940$

  The number of timing messages vs. MSEs of clock offset estimators in the presence of asymmetric Weibull delays are presented in Fig. 23. Monte Carlo Simulations: 400.

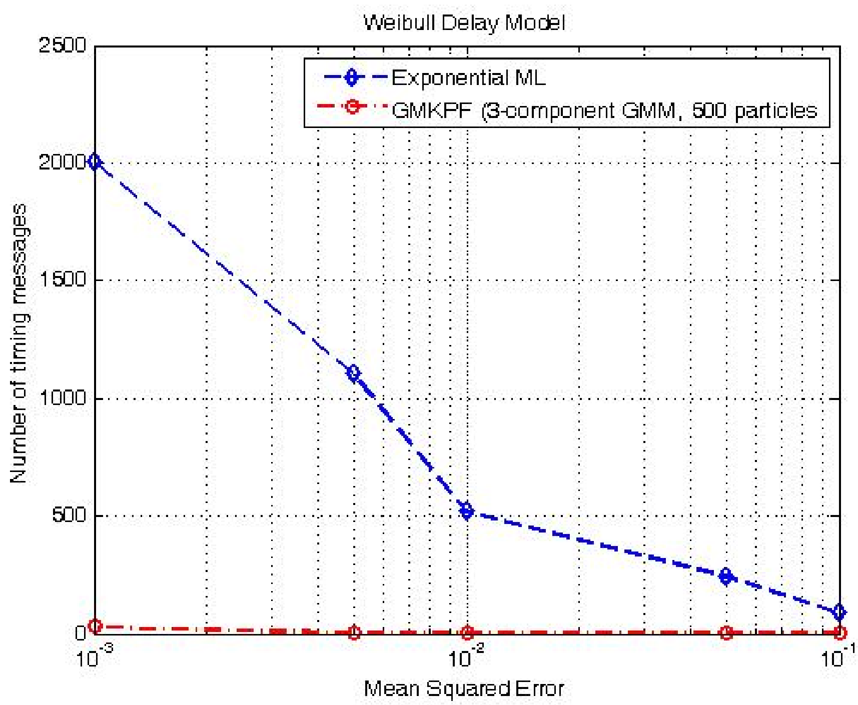

Figure 23. Number of timing messages vs. MSEs of clock offset estimators for asymmetric Weibull delays

5. CONCLUSIONS

Time synchronization is a significant component in the deployment of wireless ad-hoc networks, and a number of fundamental operations, like data fusion, power management and transmission scheduling, require accurate time synchronization. Since the conventional NTP time synchronization protocol for the Internet cannot be directly applied to wireless sensor networks and wireless airborne networks, a number of synchronization protocols have been developed in this project to meet the unique requirements of these applications.

This project developed a very general and powerful inference method for estimating the clock offset in wireless ad-hoc networks. The benefits of the proposed synchronization method are in terms of improved performance and applicability to arbitrary random delay models such as asymmetric Gaussian, asymmetric exponential, Gamma, Weibull, as well as to mixtures of these delay models. One negative aspect is the fact that analytical closed form expressions do not necessarily exist and in general it is hard to derive lower bounds in the presence of (unknown) non-Gaussian distributions. The project proposed a robust estimator based on the GMKPF that is capable of estimating the clock offset in arbitrary delay models, a result which might present applications in numerous wireless sensor networks applications with tight synchronization requirements [17], [18]. Computer simulations also show that the proposed method yields superior performance.

The proposed statistical inference mechanism is quite general and can be practically applied to any modeling problem that can be phrased in terms of a state-space representation. The additive noise can assume arbitrary distributions, and the observation equation can be linear or nonlinear. In the presence of a nonlinear observation equation, a slightly more general modeling framework in terms of particle filters might have to be adopted to handle the nonlinearities. However, in our present study, the clock observation equations are linear equations; therefore, a bank of Kalman filters was sufficient to efficiently track the unknown clock offset parameters. We would like to emphasize that the proposed statistical inference engine could be applied to more general applications such the problem of joint synchronization and localization. In this scenario, one can perform two tasks, namely clock synchronization and localization of a target using the same set of signals/data samples. Therefore, one could expect high performance and very fast algorithms to be developed using the proposed statistical inference mechanism. Furthermore, the results of this project could be extended further to address the problem of joint synchronization, localization and tracking of a moving target, or for assessing the topography of a wireless network.

6. REFERENCES

[1] K. Arvind, “Probabilistic Clock Synchronization in Distributed Systems,” *IEEE Transactions on Parallel and Distributed Systems*, 5(5):474487, May 1994.

[2] F. Cristian, “Probabilistic Clock Synchronization,” *Distributed Computing*, 3:146158, Springer-Verlag, 1989.

[3] A. Leon-Garcia, Probability *and Random Processes for Electrical Engineering*, 2nd ed. Reading, MA: Addison-Wesley, 1993.

[4] T. V. Dam, and K. Langendoen, “An adaptive energy-efficient MAC protocol for wireless sensor network,” *Proceedings of the ACM Conference on Embedded Network Sensor Systems (Sensys)*, 2003.

[5] J. Elson, L. Girod, and D. Estrin, “Fine-grained network time synchronization using reference broadcasts,” *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*, (OSDI 2002), Dec. 2002.

[6] S. Ganeriwal, R. Kumar, and M. B. Srivastava, “Timing-sync Protocol for Sensor Networks,” *Proc. of the First ACM Conference on Embedded Networked Sensor Systems*, (SenSys), Nov. 2003.

[7] Q. Li and D. Rus, “Global clock synchronization in sensor networks,” *IEEE INFOCOM*, 2004.

[8] K. Romer, “Time Synchronization in Ad Hoc Networks,” *Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing*, pp. 173182, Oct 2001.

[9] H. S. Abdel-Ghaffar, “Analysis of synchronization algorithm with time-out control over networks with exponentially symmetric delays”, *IEEE Transactions on Commun*., vol. 50, no. 10, pp. 1652-1661, Oct. 2002.

[10] I. Akyildiz et al., “Wireless sensor networks: A survey”, *Computer Networks*, vol. 38, no. 4, pp. 393-422, March 2002.

[11] B. D. Anderson and J. B. Moore, *Optimal Filtering*, Prentice-Hall, 1979.

[12] J. Elson, L. Girod, and D. Estrin, “Fine-grained network time synchronization using reference broadcasts,” *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*, (OSDI 2002), Dec. 2002.

[13] F. Pernkopf and D. Bouchaffra, “Genetic-Based EM Algorithm for Learning Gaussian Mixture Models,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, Aug. 2005.

[14] S. Ganeriwal, D. Ganesan, H. Shim, V. Tsiatsis, and M. B. Srivastava, “Estimating Clock Uncertainty for Efficient Duty-Cycling in Sensor Networks,” *ACM Sensys’05 Conference*, pp. 130-141, Nov. 2005, San Diego, CA.

[15] S. Ganeriwal, R. Kumar and M.B. Srivastava, “Timing Synch Protocol for Sensor Networks,” In *Proceedings of 1st International Conference on Embedded Network Sensor Systems (SenSys)*, ACM Press, Nov 2003.

[16] Q. Gao, K. J. Blow, and D. J. Holding, “Simple Algorithm for Improving Time Synchronization in Wireless Sensor Networks,” *Electronics Letters*, vol. 40, p. 889, 2004.

[17] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li, “Research Challenges and Applications for Underwater Sensor Networking.” In *Proceedings of the IEEE Wireless Communications and Networking Conference*, (to appear) Las Vegas, Nevada, USA, April 2006. Available online: http://www.isi.edu/~johnh/PAPERS/Heidemann06a.html.

[18] A. Syed and J. Heidemann, “Time Synchronization for High Latency Acoustic Networks,” Technical Report ISI-TR-2005-602, USC/Information Sciences Institute, April 2005. Available online: http://www.isi.edu/~johnh/PAPERS/Heidemann06a.html.

[19] S. J. Julier and J. K. Uhlmann, “A General Method for Approximating Nonlinear Transformations of Probability Distributions,” RRG, Dept. of Engineering Science, University of Oxford, Nov. 1996.

[20] A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison-Wesley, MA, 1993.

[21] R. van der Merwe and E. Wan, “Gaussian Mixture Sigma-Point Particle Filters for Sequential Probabilistic Inference in Dynamic State-Space Models”, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, April 2003.

[22] D. Mills, “Internet Time Synchronization: The Network Time Protocol; RFC 1129,” *Internet Request for Comments*, No. 1129, Oct 1989.

[23] D. L. Mills, “Internet Time Synchronization: The Network Time Protocol,” *IEEE Trans. on Communications*, vol. 39, no. 10, Oct. 1991, pp. 1481-1493.

[24] K.-L. Noh, Q. M. Chaudhari, E. Serpedin, and B. W. Suter, “Novel Clock Phase Offset and Skew Estimation Using Two-Way Timing Message Exchanges for Wireless Sensor Networks,” *IEEE Transactions on Communications*, vol. 55, no. 4, April 2007.

[25] A. Papoulis, *Probability, Random Variables and Stochastic Processes*, McGraw-Hill, 1991.

[26] D. Tulone, “Resource-efficient time synchronization for wireless sensor networks,” *Proc. DIALM-POMC Workshop on Foundations of Mobile Computing*, S. Basagni and C. A. Phillips, editors, pp. 52-50, Philadelphia, PA, Oct. 2004.

List of Symbols, Abbreviations, and Acronyms

AO = Always On

Adaptive Sync = Adaptive Synchronization

CRLB = Cramer-Rao Lower Bound

FTSP = Flooding Time Synchronization Protocol

GPS = Global Positioning System

ID = Identification

iid = independent and identically distributed

IP = Internet Protocol

MAC = Medium Access Control

ML = Maximum Likelihood

MLE = Maximum Likelihood Estimator

MSc = Master of Science

MSE = Mean Square Error

NTP = Network Time Protocol

PBS = Pairwise Broadcast Synchronization

PDF = Probability Density Function

PhD = Doctor of Philosophy

RBS = Reference Broadcast Synchronization

RF = Radio Frequency

ROS = Receiver Only Synchronization

RRS = Receiver Receiver Synchronization

RV = Random Variable

SI = Sensor Initiated

SRS = Sender Receiver Synchronization

TDMA = Time Division Multiple Access

TDP = Time Diffusion Protocol

TPSN = Time Protocol for Synchronization of Sensor Networks

UAV = Unmanned Aerial Vehicles

UDP = Universal Data Protocol

WAN = Wireless Airborne Network

WLAN = Wireless Local Area Network

wrt = with respect to

WSN = Wireless Sensor Network